

DebianにおけるSDカードへの書き込み回数の低減方法

2019年03月08日
東芝メモリ株式会社
メモリ技術研究所
システム技術研究開発センター
山田 真大



BiCS FLASHTM

- 01 背景
- 02 課題
- 03 アプローチ
- 04 評価
- 05 まとめと今後の課題

組み込み機器ではBoot DeviceとしてSDカードを使う機会が多い

典型的な組み込み機器

- 例：Raspberry PiはSDカードにraspbianをインストールして起動
- SDカードには書き込み回数に制限がある
- Boot deviceがSDカードしかない場合、SD Cardの寿命が尽きればそのボードは起動しない

Linux Distributionを使う場合

- ユーザが認識していない（意図していない）書き込みが多く存在
- 上記を考慮にいれていなければ、SDカードの寿命は想定よりも少ない

ユーザが認識していない書込みの例

- nginx.serviceが稼働している間、Webサーバにアクセスがあったときや問題が発生した際、access.logやerror.logにそのlogが記録される

```
$ tree /var/log/nginx
/var/log/nginx
├── access.log
└── error.log

0 directories, 2 files
```

- (極端に言えば) 外部アクセスによって組込み機器(SDカード)は消耗

意図していない書き込みを極力減らし、寿命を延ばす対策をとりたい

極端な対策（SDカードへのすべての書き込みをシャットアウト）：

- rootfsをread only mountにして、一切書き込みをしない
- NFS mountにより、書き込みは全て外部ストレージで行う

少し柔軟な対策(SDカードへの一部の書き込みをシャットアウト)：

- 書き込み頻度が多く、かつ、重要性の低い一部のディレクトリやファイルに対する書き込みを揮発化・無効化
- システムが稼働中にアクセスできればよい。rebootを行った場合は、それらが消えてしまっても問題ないという前提条件

本発表の仮定

- CPUはx86(64bit)。評価環境ではVMを利用する
- OSはDebian9.5/9.6, kernelはv4.9
- rootfsのフォーマットとしてext4を使う
- RootFSはSDカードに格納する
- 起動時にSDカードをrootfsとしてmountして利用する
- 書込みとして使用できる他の外部ストレージはないという想定

シャットアウトできそうなSDカードへの書込み要因を5つに分類

- 課題1 systemd serviceによるログファイルへの定期的な書込み
- 課題2 cacheファイル
- 課題3 ファイルへのアクセス時間の記録
- 課題4 スワップファイル
- 課題5 上記以外の未知の書込みを行うプロセスの発見

課題1 systemd serviceによるログファイルへの定期的な書込み

- システムが起動した後、多くのsystemd serviceが同時に起動し、定常的に動作を続ける。ユーザが意図していない場合でもデフォルト設定により起動するサービスは多くある
- 下記コマンドでデフォルトで起動しているsystemd serviceの確認が可能

```
$ systemctl list-unit-files | grep enabled
```

- これらのサービスの中には、その実行時のログを/var/log/の下にあるファイルに保存するものがある

課題2 cacheファイル

- ある特定の処理を高速化するためdisk上にcacheを残しておくプログラムが存在（e.g. ネットワークアクセスを減らすためのキャッシュファイル）
- キャッシュとして残しておくファイルは、そのサイズが大きくなる傾向にあり
⇒SDカードの書き込み負荷
- パッケージ管理システムであるaptは、キャッシュの利用をしている典型的なプログラムである。/var/cache/aptには、これまでインストールを行ったdebファイルが書込まれる

課題3 ファイルへのアクセス時間の記録

ext4では各ファイル・ディレクトリに対して3つのメタデータを用意

- Access time、Modify time、Change time
- Access timeは、ファイル・ディレクトリに対して読み書きを行った時刻

ext4のmount optionはデフォルトでrelatime

- ファイルの読み込み処理であったとしても、過去24H以内のアクセスがなかった場合に、Access timeを更新するため、ディスクへの書き込み処理が発生

課題4 スワップファイル

- スワップ=メモリ不足で動作不能になるまえに長時間使用されていないメモリをディスク上に退避し、メモリの空き容量を増やす
- 搭載しているメモリのサイズやシステム上で動作させるアプリケーションによっては、頻繁にスワップが発生する⇒SDカードへの書込み負荷
- Debianではswapの設定が/etc/fstabに記述、起動時にsystemd-remount-fs.serviceがswap領域のマウントを行いスワップを有効化

課題5 課題1~4以外の未知の書込みを行うプロセスの発見

課題1~4以外にも定期的、突発的に書込みを行うプロセスは存在

- 例：cronやsystemdのtimer処理
- 例：kernel threadが行うext4のjournal処理

これらが無効化してよいかの判断する前に、そもそも“誰”が“どこ”に書込みをしているかを発見するための手段が必要

3種類のアプローチ

アプローチ1 tmpfsを使ってディレクトリ、ファイルを揮発化（課題 1 , 2）

アプローチ2 機能の無効化（課題 3 , 4）

アプローチ3 "だれ"が"どこ"にアクセスしているかを検出（課題 5）

アプローチ1 tmpfsによるディレクトリ、ファイルの揮発化(1)

以下2つのsystemd serviceを使って実現

- systemd-remount-fs.service(fstab)でtmpfs作成
- systemd-tmpfiles-setup.serviceで既存ディレクトリ、ファイル置き換え

揮発化対象にできるディレクトリは、いろいろある

- 例： /var/lock、/var/run、/var/tmp、/tmp、/run/systemd、...

今回は話を単純にするため、/var/logと/var/cache/aptを揮発化

アプローチ1 tmpfsによるディレクトリ、ファイルの揮発化(2)

fstabで/var/volatileをtmpfsとしてマウント

```
$ cat /etc/fstab
#<file system> <mount point> <type> <options> <dump> <pass>
...
tmpfs      /var/volatile tmpfs defaults      0    0
```

systemd-tmpfiles-setup.serviceの設定で2つのディレクトリを揮発化

```
$ cat /etc/tmpfiles.d/00_core.conf
d /var/volatile/log 0755 root root -
L+ /var/log 0755 root root - /var/volatile/log
d /var/volatile/cache/apt 0755 root root -
L+ /var/cache/apt 0755 root root - /var/volatile/cache/apt
```

アプローチ1 tmpfsによるディレクトリ、ファイルの揮発化(3)

本アプローチを使うことで起きる問題点

- /var/logを揮発化すると動作しなくなるパッケージが存在
- 例：nginxは、systemd service起動時に/var/log/nginx以下が存在することを前提。存在しない場合、起動に失敗

```
Sep 15 00:58:17 debian09 systemd[1]: Starting A high performance web server and a reverse proxy server...
Sep 15 00:58:17 debian09 nginx[4161]: nginx: [alert] could not open error log file: open() "/var/log/nginx/error.log" failed
Sep 15 00:58:17 debian09 nginx[4161]: 2018/09/15 00:58:17 [emerg] 4161#4161: open() "/var/log/nginx/access.log" failed (2:
Sep 15 00:58:17 debian09 nginx[4161]: nginx: configuration file /etc/nginx/nginx.conf test failed
Sep 15 00:58:17 debian09 systemd[1]: nginx.service: Control process exited, code=exited status=1
Sep 15 00:58:17 debian09 systemd[1]: Failed to start A high performance web server and a reverse proxy server.
Sep 15 00:58:17 debian09 systemd[1]: nginx.service: Unit entered failed state.
Sep 15 00:58:17 debian09 systemd[1]: nginx.service: Failed with result 'exit-code'.
```


アプローチ1 tmpfsによるディレクトリ、ファイルの揮発化(4)

対策

- システム起動時に/var/log/nginx以下を作成

```
$ cat /etc/tmpfiles.d/50_nginx.conf  
d /var/log/nginx 0755 root adm - none  
f /var/log/nginx/access.log 0640 www-data adm - none  
f /var/log/nginx/error.log 0640 www-data adm - none
```

nginxのように動作しなくなるパッケージはDebianにいくつもあるので注意

- 参考としていくつかのパッケージを列挙 (Appendix. A)

アプローチ1 tmpfsによるディレクトリ、ファイルの揮発化(5)

本アプローチの注意点

- メモリ使用量。ログファイルを書き込む分だけメモリが使用される
- Tmpfsはデフォルトで物理メモリの半分まで使用

size: The limit of allocated bytes for this tmpfs instance. **The default is half of your physical RAM without swap.** If you oversize your tmpfs instances the machine will deadlock since the OOM handler will not be able to free that memory.

- ログを書き込むパッケージには、logrotateの設定ファイルがあるかどうか注意。また、設定も適切な頻度かどうか要検討

アプローチ2 機能の無効化(1)

fstabでrootディレクトリをnoatime mount

```
$ cat /etc/fstab
#<file system> <mount point> <type> <options>          <dump> <pass>
...
/dev/root    /           ext4    rw,noatime,data=ordered 0    0
```

本アプローチの注意点

- いくつかのパッケージはatimeが正しく更新されることを前提に動作
- 例：mutt（メールソフト）
- 電源ON ~ remountまでの間は、atimeが更新される可能性あり

アプローチ2 機能の無効化(2)

fstabでswapの無効化

```
$ cat /etc/fstab
...
# swap was on /dev/sda5 during installation
#UUID=7f126736-bc08-4569-a3d0-049e1f09ebd7 none          swap    sw          0        0
```

本アプローチの注意点

- メモリ使用量。アプローチ1と合わせて、最大でどれくらいメモリを使うかあらかじめ検討する必要あり。

アプローチ3 "だれ"が"どこ"にアクセスしているかを検出

今回はFanotifyを使う

- fanotifyはファイルシステムのイベントを監視するための仕組み
 - <sys/fanotify.h>をincludeすることで各fanotify APIを使用
 - fanotify APIを使用して、あるマウントポイントの下にあるファイルに対する操作を監視し、その操作を行ったプロセスのIDを知ることができる。
 - exampleプログラム： <https://manpages.debian.org/stretch/manpages/fanotify.7.en.html>
 - 書込み監視 + pidを出力できるようにexample修正 (Appendix. B)
- (参考) Bcc-toolsを使っても同様のことができる(Appendix. C)

Diskに対する書き込み要求を確認する方法

Sysfsのblockデバイスのstatファイルにあるwrite io数をチェック

- /dev/sdaがBoot deviceの場合

```
$ cat /sys/block/sda/stat  
11976 2145 680990 12200 212 383 5320 340 0 4704 12532
```

- 第5パラメータが書き込みに関するbio要求の数

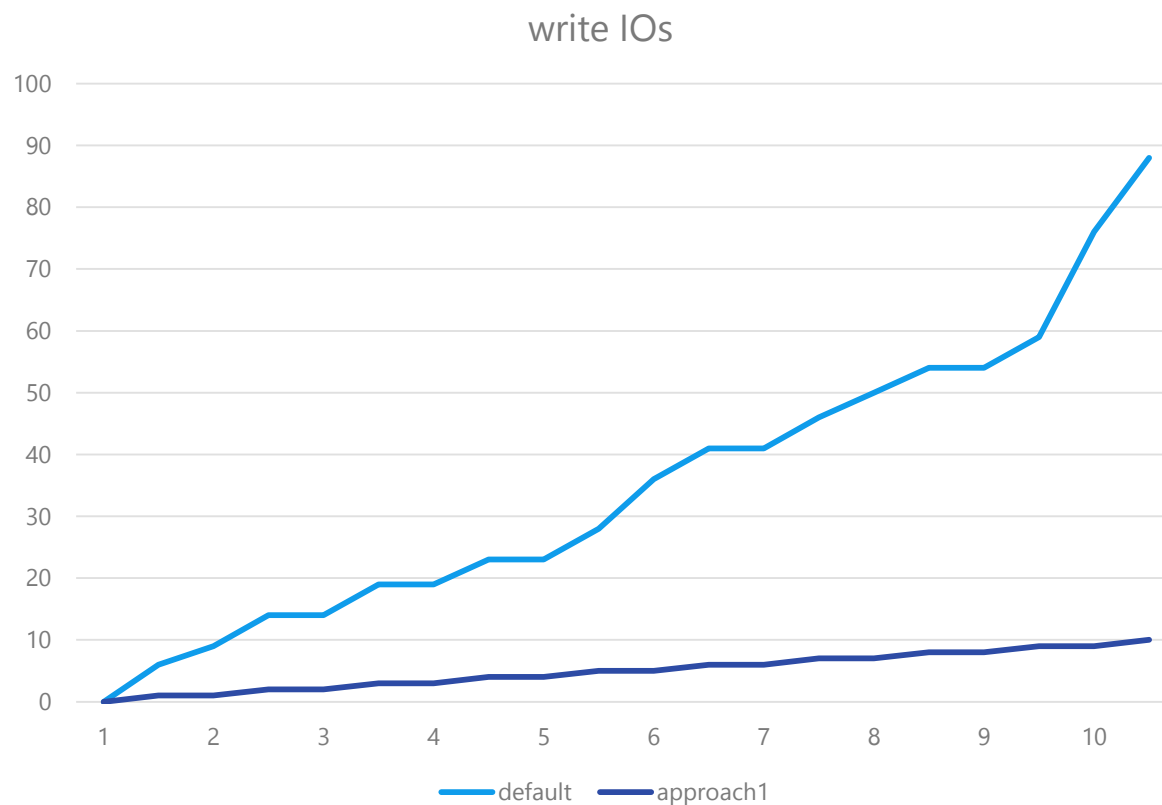
Name	units	description
----	-----	-----
read I/Os	requests	number of read I/Os processed
read merges	requests	number of read I/Os merged with in-queue I/O
read sectors	sectors	number of sectors read
read ticks	milliseconds	total wait time for read requests
write I/Os	requests	number of write I/Os processed

引用 : <https://www.kernel.org/doc/Documentation/block/stat.txt>

課題1 (/var/logへの書込み)+アプローチ1(揮発化)の評価結果

以下の手順を10回繰り返す

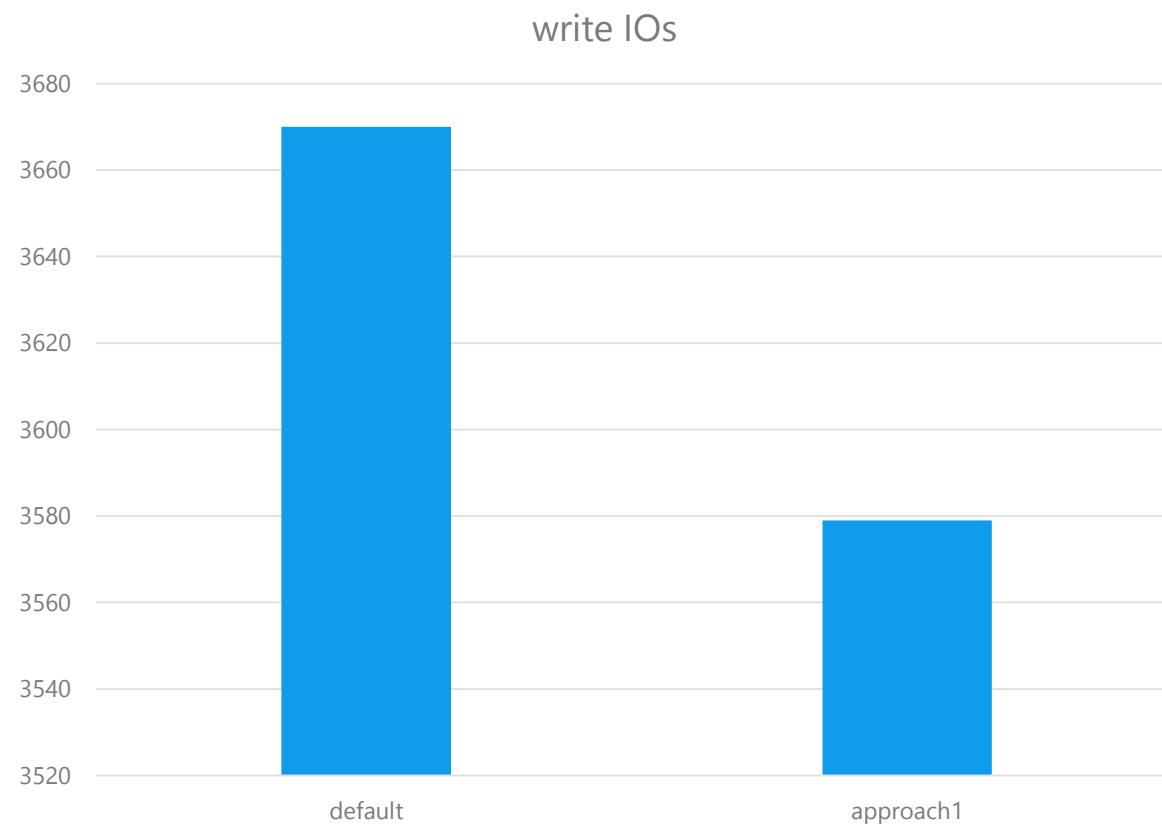
- statファイルを確認
- curlコマンドによるnginxアクセス
- syncコマンドの実行
- statファイルの確認
- 60秒間のスリープ



課題2(cache file)+アプローチ1(揮発化)の評価結果

以下の手順を実施

- statファイルを確認
- aptコマンドでvimのインストール
- syncコマンドの実行
- statファイルの確認

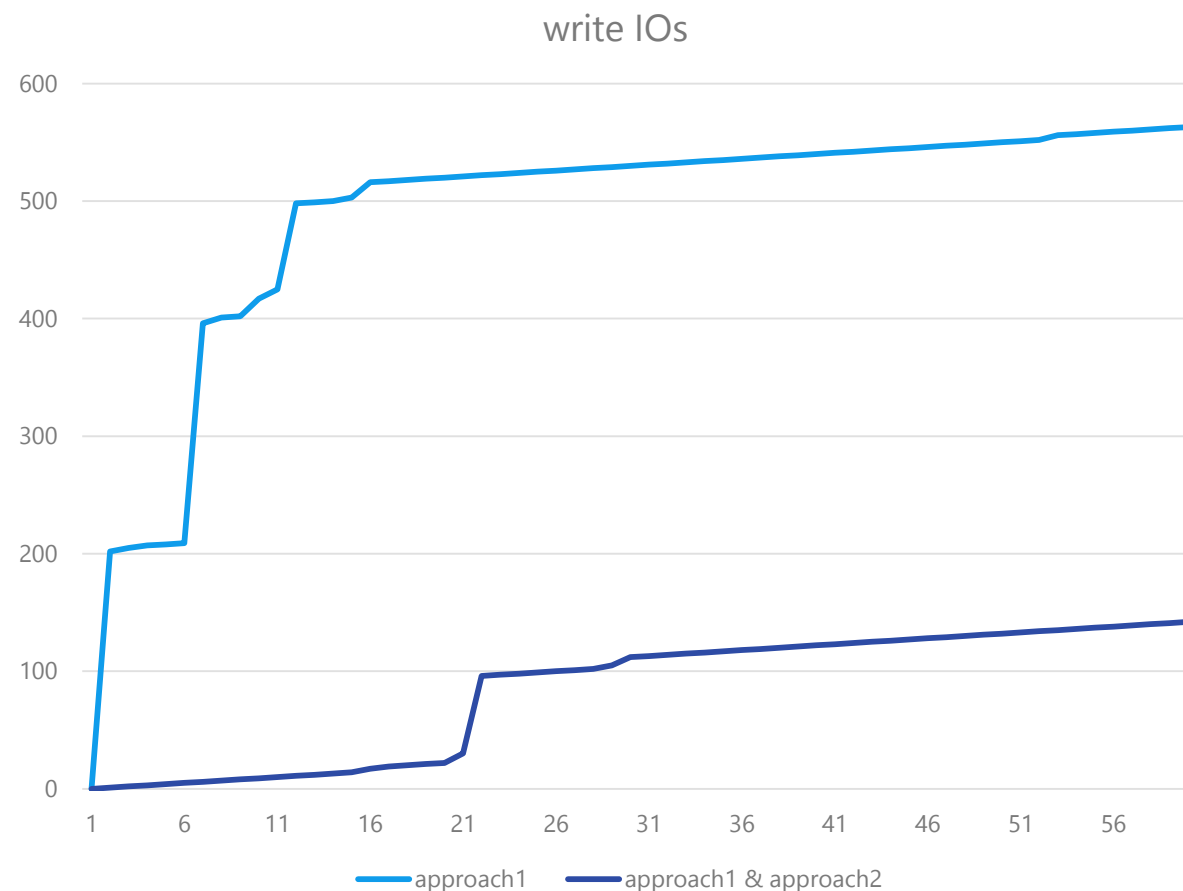


課題3(access timeの更新)+アプローチ2(無効化)の評価結果

approach1(揮発化)を適用済みにして、評価中の
Systemd serviceによる/var/logへの書き込み分を省く

以下の手順を実施

- システム時刻を24H進める
- syncコマンドの実行
- 以下を60回ループ(1時間分の測定)
- syncコマンドの実行
- statファイルによりIOの確認
- 1分sleep



課題4(スワップ)+アプローチ2(無効化)の評価結果

- 適用前

```
# cat /proc/swaps
Filename                Type      Size  Used  Priority
/dev/sda5               partition 2095100 0      -1
```

- 適用後

```
# cat /proc/swaps
Filename                Type      Size  Used  Priority
```

課題5(書込み検出)+アプローチ3(監視プログラム)の評価結果

- fanotifyでルートディレクトリ"/"以下を1時間監視
- アプローチ1,2適用済みの環境（既知の書き込みを検出しないため）
- 結果の一部を掲載(Appendix. D)

```
# fanotify_example /  
...  
FAN_OPEN_PERM: File /var/lib/upower/history-time-empty-50.dat.VTEKPZ (657)  
FAN_MODIFY: File /var/lib/upower/history-time-empty-50.dat.VTEKPZ (657)  
FAN_CLOSE_WRITE: File /var/lib/upower/history-time-empty-50.dat (657)  
...
```

- PID 657を調べるとupower.serviceによるものと判明
アプローチ1かアプローチ2を適用すれば、さらにdiskアクセス低減（要検討）

まとめと今後の課題

まとめ

- SDカードへの不要な書き込みを減らすため、設定ファイルによるファイルやディレクトリの揮発化や特定機能の無効化で、diskアクセスを低減
- 書き込みの監視用プログラムで、認識していない書き込み処理を検出

今後の課題

- 揮発化により動作しなくなるパッケージへの対応をより簡単にしたい
- Kdumpを導入している場合、panic発生で/var/log以下の内容も消えてしまう。ダンプファイルから/var/log以下の内容を参照するのは難しい？



BiCS FLASHTM

Appendix

Appendix A. tmpfs化するときに注意すべきパッケージ

Package / command	Directory / file
openssh-server	/var/run/sshd
nginx	/var/log/nginx /var/log/nginx/access.log /var/log/nginx/error.log
Uwsgi	/var/log/uwsgi /var/log/uwsgi/app /var/run/uwsgi
audit	/var/log/audit
dpkg	/var/log/dpkg
libvirt	/var/log/libvirt/qemu /var/log/libvirt/uml /var/log/libvirt/lxc
apache2	/var/log/apache2 /var/run/apache2
sysstat	/var/log/sa
Exim	/var/log/exim4

Appendix B. fanotify exampleの修正パッチ

```
$ diff -up fanotify_example.c fanotify_example_write.c
--- fanotify_example.c 2018-09-20 07:21:17.000000000 +0900
+++ fanotify_example_write.c 2018-09-20 13:18:34.328000000 +0900
@@ -78,6 +78,9 @@ handle_events(int fd)
     if (metadata->mask & FAN_CLOSE_WRITE)
         printf("FAN_CLOSE_WRITE: ");

+    if (metadata->mask & FAN_MODIFY)
+        printf("FAN_MODIFY: ");
+
     /* Retrieve and print pathname of the accessed file */

     snprintf(procd_path, sizeof(procd_path),
@@ -90,7 +93,7 @@ handle_events(int fd)
     }

     path[path_len] = '\0';
-    printf("File %s\n", path);
+    printf("File %s (%d)\n", path, metadata->pid);

     /* Close the file descriptor of the event */

@@ -136,7 +139,7 @@ main(int argc, char *argv[])
     file descriptor */

     if (fanotify_mark(fd, FAN_MARK_ADD | FAN_MARK_MOUNT,
-        FAN_OPEN_PERM | FAN_CLOSE_WRITE, AT_FDCWD,
+        FAN_OPEN_PERM | FAN_CLOSE_WRITE | FAN_MODIFY, AT_FDCWD,
         argv[1]) == -1) {
         perror("fanotify_mark");
         exit(EXIT_FAILURE);
     }
 }
```


Bcc-tools (<https://github.com/iovisor/bcc>) 付属のexampleであるbiosnoopやfiletopを使う

- bcc-toolsのbiosnoop

```
TIME(s)  COMM      PID  DISK  T  SECTOR  BYTES  LAT(ms)
0.000000000 ?      0      R  -1      8      0.49
2.015770000 ?      0      R  -1      8      0.25
2.271162000 jbd2/sda1-8 152  sda   W 80805472 16384  0.42
2.275804000 jbd2/sda1-8 152  sda   W 80805504 4096   0.23
4.030423000 ?      0      R  -1      8      0.17
6.048356000 ?      0      R  -1      8      0.45
8.064035000 ?      0      R  -1      8      0.23
8.158926000 kworker/u2:0 17059 sda   W 2048   4096   0.22
8.159107000 kworker/u2:0 17059 sda   W 2080   4096   0.35
```

- bcc-toolsのfiletop

```
03:46:23 loadavg: 0.17 0.10 0.16 2/321 19463

TID  COMM      READS  WRITES R_Kb  W_Kb  T FILE
19463 clear      2      0      8      0      R xterm-256color
19452 python     2      0      2      0      R loadavg
19463 clear      1      0      0      0      R libtinfo.so.5.9
19463 clear      1      0      0      0      R libc-2.24.so
19463 python     3      0      0      0      R clear
19463 python     2      0      0      0      R ld-2.24.so
```

Appendix D. fanotify_exampleによる/以下の監視結果

```
# fanotify_example /
...
FAN_MODIFY: File /var/lib/upower/history-rate-50.dat.W2VKPZ (657)
FAN_CLOSE_WRITE: File /var/lib/upower/history-rate-50.dat (657)
FAN_OPEN_PERM: File /var/lib/upower/history-charge-50.dat.VUCKPZ (657)
FAN_MODIFY: File /var/lib/upower/history-charge-50.dat.VUCKPZ (657)
FAN_CLOSE_WRITE: File /var/lib/upower/history-charge-50.dat (657)
FAN_OPEN_PERM: File /var/lib/upower/history-time-full-50.dat.DDHKPZ (657)
FAN_MODIFY: File /var/lib/upower/history-time-full-50.dat.DDHKPZ (657)
FAN_CLOSE_WRITE: File /var/lib/upower/history-time-full-50.dat (657)
FAN_OPEN_PERM: File /var/lib/upower/history-time-empty-50.dat.VTEKPZ (657)
FAN_MODIFY: File /var/lib/upower/history-time-empty-50.dat.VTEKPZ (657)
FAN_CLOSE_WRITE: File /var/lib/upower/history-time-empty-50.dat (657)
...
FAN_OPEN_PERM: File /var/lib/systemd/timers/stamp-anacron.timer (1)
FAN_CLOSE_WRITE: File /var/lib/systemd/timers/stamp-anacron.timer (1)
...
FAN_OPEN_PERM: File /etc/anacrontab (1343)
FAN_OPEN_PERM: File /var/spool/anacron/cron.daily (1343)
FAN_CLOSE_WRITE: File /var/spool/anacron/cron.daily (1343)
FAN_OPEN_PERM: File /var/spool/anacron/cron.weekly (1343)
FAN_CLOSE_WRITE: File /var/spool/anacron/cron.weekly (1343)
FAN_OPEN_PERM: File /var/spool/anacron/cron.monthly (1343)
FAN_CLOSE_WRITE: File /var/spool/anacron/cron.monthly (1343)
...
FAN_CLOSE_WRITE: File /tmp/tmpfdrUyvr (deleted) (1356)
FAN_CLOSE_WRITE: File /tmp/tmpfdrUyvr (deleted) (1356)
...
```

- <https://wiki.debian.org/SSDOptimization>
- <http://blog.nunosenica.com/reduce-write-operations-to-sd-card-with-raspbian/>
- <https://www.makeuseof.com/tag/extend-life-raspberry-pis-sd-card/>
- <https://www.kernel.org/doc/Documentation/filesystems/tmpfs.txt>
- <https://www.kernel.org/doc/Documentation/block/stat.txt>
- <http://hallard.me/raspberry-pi-read-only/>